Testing of Haar-like feature in Region of Interest Detection for Automated Target Recognition (ATR) System

Yuhan Zhang

Mentor: Dr. Thomas Lu
Jet Propulsion Laboratory, California Institute of Technology

This Cooperative Education Report has been submitted to (Dr. Thomas Lu) and has been approved for release and submission to Dr. Francis X. Flores, Cooperative Education Director, Cal Poly, Pomona **during the 9th week of the quarter.**

Signed _____ Dated _____

Objectives:

The objectives of this project were to develop a ROI (Region of Interest) detector using Haar-like feature similar to the face detection in Intel's OpenCV library, implement it in Matlab code, and test the performance of the new ROI detector against the existing ROI detector that uses Optimal Trade-off Maximum Average Correlation Height filter (OTMACH).

The ROI detector included 3 parts: 1, Automated Haar-like feature selection in finding a small set of the most relevant Haar-like features for detecting ROIs that contained a target. 2, Having the small set of Haar-like features from the last step, a neural network needed to be trained to recognize ROIs with targets by taking the Haar-like features as inputs. 3, using the trained neural network from the last step, a filtering method needed to be developed to process the neural network responses into a small set of regions of interests. This needed to be coded in Matlab. All the 3 parts needed to be coded in Matlab. The parameters in the detector needed to be trained by machine learning and tested with specific datasets.

Since OpenCV library and Haar-like feature were not available in Matlab, the Haar-like feature calculation needed to be implemented in Matlab. The codes for Adaptive Boosting and max/min filters in Matlab could to be found from the Internet but needed to be integrated to serve the purpose of this project.

The performance of the new detector was tested by comparing the accuracy and the speed of the new detector against the existing OTMACH detector. The speed was referred as the average speed to find the regions of interests in an image. The accuracy was measured by the number of false positives (false alarms) at the same detection rate between the two detectors.

Executive Summary:

JPL was researching on an Automated Target Recognition (ATR) system that could locate objects of interest in an image. There had existed an ATR system using optical correlation-based filter called OTMACH filter at the first stage to detect regions of interests (ROI). My task at this internship was to implement and test an alternative algorithm (called "Haar-like" feature) to serve as the ROI detector for potential improvement in detection accuracy.

Haar-like feature measured the contrast between the total intensities in two rectangular regions in an image. This process was slow in the brute force approach, but could be completed fast in *O(1)* time using a technique called "summed area table". Considering all possible choices of pairs of sampling rectangles within a ROI to generate Haar-like features, the pairs of rectangles distinguish data of different classes the best were picked using AdaBoost with a given set of training examples. This small set of rectangle pairs was called "Haar-like kernel". This Haar-like kernel was used to extract feature vectors for every ROI location on the image. An artificial neural network was trained by examples to recognize the ROIs that contain a target. The neural network together with Haar-like kernel could convert the image into an image of neural network responses. A filtering method was used to analyze the image of neural network responses to receive ROIs. Three filtering methods were tested: 1, a combination of a median filter and a max filter; 2, a filter detecting the center of mass for large clusters of high neural network responses; 3, a filter detecting the center of the mass for large clusters of high neural network responses with a min filter.

Two sonar image datasets and four boat image datasets were experimented. With the third filtering method, the new detector achieved over 90% of catch rate with less than 40 false alarms per image on the sonar dataset. The false alarm rate was twice as smaller comparing to the existing OTMACH filter at the same accuracy in the two sonar datasets. For the boat datasets, the false alarm rate of the new detector was closely under to OTMACH filter, while still being reasonable in accuracy.

Description of Co-op Position and Activities:

Dr. Thomas Lu (my mentor) was leading a research in target recognition system and developing a set of advanced computer vision algorithms that automatically recognize and locate targets in an image. This project mostly aims at detection of unauthorized boats from an unmanned patrol boat and detection of underwater mine. This technology could potentially be applied for the detection of any object in digital images. Prior to this work, there had existed an automatic target recognition (ATR) system in JPL using a combination of OTMACH filter and artificial neural network [2]. The ATR system recognized objects of interest in an image by a trained artificial neural network and a trained OTMACH filter by a set of training examples (images). The system was implemented in Matlab.

My task in this co-op position was to explore, implement and test an alternative approach that could potentially replace OTMACH filter with a more generalized and more accurate solution (faster, too, if possible). One promising alternative approach was the face detection algorithm in Intel's OpenCV library using Haar-like feature. However, since OpenCV was in C/C++, to adapt Haar-like feature to the existing ATR system, this functionality needed to be implemented in Matlab. 30% of my time at this internship was spent on implementing the Haar-like feature extraction and summed area table, and integrating it with the neural network for training and classification. 30% of the time was spent on looking for Matlab code for Adaptive Boosting and adopting it for feature selection in finding a smaller set of Haar-like features. 30% of the time was spent on correcting and debugging the application, training the neural network and the Haar-like kernel for different datasets, and running the classification algorithm to measure the performance, as well as understanding the problems and tuning up the code, reading of related documents, proposing and testing different solutions. 10% of the time was spent of discussing the results with my mentor and my colleague, as well as preparing for presentation slides and reports.

Description of Project completed and Data:

Implementation of Haar-like feature extractor

Haar-like feature:

Haar-like feature is currently being used in the computer vision library OpenCV (developed by Intel) for face detection in an image [1]. In general, Haar-like feature measures the contrast between the total intensities in the two boxes in an image. This feature is called a "Haar-like feature". There exists a large set of such features with different choices of boxes in a ROI: across various sizes, at various locations, and with different pairing. Each Haar-like feature consists two rectangles: one outer rectangle and one inner rectangle. A Haar-like feature is calculated using: [3]

$$feature = w_2 * RectSum_1(I) - w_1 * RectSum_2(I) \quad (1)$$

The weight w_1 and w_2 are the number of pixels in the two boxes. The function $RectSum_i(I)$ gives the sum of the intensity values within the rectangle i in image I. This feature value will be 0 if the average intensity inside the two rectangles are the same. It will be negative or positive, indicating the significance of contrast.

Summed area table

To calculate the total intensity within a rectanglar region in an image, the brute force approach is very inefficient. To make Haar-like feature to be extracted in real time, Viola and Jones employed a technique called "summed-area table", which was developed by Crow in 1984. Each entry at (x,y) location in the summed-area table is the sum of the values within the rectangle from $(0,0)$ to (x,y). The value of each entry can be constructed based on the three previous neighboring entries and the current intensity using dynamic programming [4]:

$$sat(x,y) = i(x,y) + sat(x-1, y) + sat(x,y-1) - sat(x-1, y-1) \quad (2)$$

Once the summed area table is constructed, the total intensity of any rectangle *(x1, y1, x2, y2)* can be calculated using the following formula within O(*1*) time using [5]:

$$sum\ (x1, y1, x2, y2) = sat\ (x2, y2) + sat\ (x1,y1) - sat\ (x1,y2) - sat\ (x2, y1) \quad (3)$$

Haar-like feature in ATR

In this experiment, the Haar-like feature is constructed following the same concept of comparing the contrast between the sums of intensity within rectangles. However, instead of having a fixed set of pairs of rectangles as OpenCV, all possible combinations of rectangle pairs within a ROI are examined and prioritized.

The implementation details have been omitted in this report. It basically follows equation (1) but with some optimization to avoid unnecessary repetition in calculation.

Implementation of AdaBoost feature selector and construction of decision image

Adaptive-Boosting of weak classifiers

The idea of Adaptive-Boosting (AdaBoost) of weak classifiers was originally formulated by Yoav Freund and Robert Schapire[8]. The idea is based on a notion that the linear combination of the results from the weak classifiers can make a stronger classifier. A weak classifier can be a threshold classifier that classifies only according to one feature value. If the data in the given dimension cannot well separate the two classes of objects by a single threshold, the weak classifier will misclassify some data points. However, given enough of such kinds of weak classifiers, the classifiers can catch the misses of each other and make into a stronger classifier [5].

The details about the mechanism of AdaBoost have been omitted in this report. In this experiment, the AdaBoost library was downloaded from Matlab central at http://www.mathworks.com, which could be easily reached by a Google search using keywords “adaboost matlab”.

Haar-like kernel training

Since there are many possible pairs of rectangles existing in a ROI to calculate features, it is necessary to reduce it only to a small set of features that can well separate the ROIs that contain object from the ROIs that contain no object. This smaller set of features of rectangle pairs is called "Haar-like kernel" in this experiment.

Training data are randomly selected as locations in the images. Once having the training data, all possible pairs of rectangles will be used to extract Haar-like features from these locations. Adaptive-Boosting (AdaBoost) is then performed on these extracted features to pick the dimensions that can well separate the training data. The top few dimensions are picked, and the rest is assumed to be the noise.

Neural network training and classification

Once having the Haar-like detection kernel constructed, a random sampling of training data happens again on the images to collect the new training data with fewer dimensions. Neural network training is performed with the new training data to receive a neural network classifier.

Filtering into detection image

The trained neural network is moved through the image at all possible locations on the image. Haar-like kernel is applied to retrieve a feature vector from each location. The trained neural network assigns each pixel location with a neural network response corresponding to the feature vector. The regions that contain a target usually have a cluster of high reponses. Taking a threshold against the responses will result many detections around the same region. To reduce the number of detections, a max filter is applied to the neural network response image to only keep the peak response within a neighborhood. Notice that there could also be random matches with very high response sparkled around the image to cause false alarms. To reduce those sharp peaks, a median filter is applied to smooth away the random sparkles that occur by chance and reserve only the clustered large values.

Besides using median filter and local max filter, an alternative approach is to measure the displacement of the mass center in the neural network response image. Since a target usually causes the points nearby to have a cloud of high responses, it will move the detection closer to the target if the mass

center of the cloud can be measured. The calculation for the center of the mass is omitted. In general, it can be calculated efficiently using summed area table technique, as mentioned about. The displacement between the center of the mass and the center of the ROI is calculated using city block distance (which is $dx + dy$) for simplicity. The second filtering method considers only the locations with small shifting in the center of mass.

The third filtering method serves as a combination of above two filtering method. First, the mass center displacements are calculated at all locations in an image. Then, a threshold is applied to only consider the locations with small displacements. This threshold is intentionally kept not to be too low to allow more mass centers to appear even at small clusters of high values. Since there are still too many mass centers appearing in a cluster of high values, a local min filter is applied to get the mass center with the smallest displacement. And these locations are counted as the center locations of ROIs.

Results:

Two sonar images datasets and four boat image datasets were tested with the two ROI detection methods (The new Haar-like detector vs. OTMACH filter) for comparison. On average, the new detector was slower than OTMACH filter (3.8 seconds per image vs. 2.5 seconds per image, based on the performance measurement on the sonar image datasets).

Three filtering techniques for the new detector were tested. The third filtering method is more superior over the other two (see table 1). In the two sonar datasets, the new detector with mass center displacement reduced more than half of the false alarms that otherwise would be considered by OTMACH filter. With the boat datasets, the new ROI detector performed closely under OTMACH filter while still being reasonable. This was because the targets in the boat dataset were small in the images, which was harder to be defined by Haar-like features (because Haar-like features considered total intensity in a large area). OTMACH in general works well when dealing with clear images, while the new Haar-like detector tolerates noisy images.

Table 1: Comparing the Haar-like feature ROI detector with OTMACH, with three different filtering methods.

Dataset \ Methods	Performance comparing to OTMACH (number of false positives per image at the same accuracy)					
	Median and max filter vs. OTMACH		Mass center shift vs. OTMACH		Mass center shift and min filter vs. OTMACH	
	Haarlike (new)	OTMACH	Haarlike (new)	OTMACH	Haarlike (new)	OTMACH
Sonar_New 64 images, 132 targets	84.4 false positives at 95% accuracy	80.4 false positives at 95% accuracy	34.7 false positives at 84% accuracy	46 false positives at 84% accuracy	39.3 false positives at 90.8% accuracy	80 false positives at 90.8% accuracy
Sonar2_New 131 images, 184 targets	81 false positives at 90% accuracy	74 false positives at 90% accuracy	21 false positives at 80% accuracy	31 false positives at 80% accuracy	22 false positives at 90.8% accuracy	79 false positives at 90.8% accuracy

Table 2: Comparing the Haar-like feature ROI detector with OTMACH, with Mass center shift and min filter for Haar-like feature ROI reduction.

Dataset \ Methods	Haarlike and Mass center shift with min filter vs. OTMACH (number of false positives per image at the same accuracy)		
	Haar-like (new)	OTMACH	Improved
Sonar_New: 64 images, 132 targets	39.3 false positives at 90.8% accuracy	80 false positives at 90.8% accuracy	Yes. Saves 40.7 false alarms
Sonar2_New: 131 images, 184 targets	22 false positives at 90.8% accuracy	79 false positives at 90.8% accuracy	Yes. Saves 55 false alarms
Oct 22Data_12_04_40: 450 images, 223 targets	3.6 false positives at 87.7% accuracy	2 false positives at 87.7% accuracy	No. Adds 1.6 false alarms
Oct 22Data_11_48_41: 253 images, 88 targets	10.2 false positives at 80.7% accuracy	8 false positives at 80.7% % accuracy	No. Adds 1.9 false alarms
2009-11-05-Thu-09-26-46_camera0: 259 images, 106 targets	2.8 false positives at 98.9% accuracy	1 false positives at 98.9% accuracy	No. Adds 1.8 false alarms
2009-11-05-Thu-11-28-54: 150 images, 127 targets	1.9 false positives at 68.2% accuracy	1 false positives at 68.2% accuracy	No. Adds 0.9 false alarms

Suggestion for Further work:

Though the new ROI detector outperformed JPL's existing OTMACH by reducing nearly half of the false positives in the sonar image datasets, the speed of this new ROI detector was very slow comparing to OTMACH filter. However, this could be easily accelerated by reducing the sampling rate of Haar-like feature. A min filter was applied over the image at all locations. The processing time could be reduced by performing the min filter only at the locations that have large neural network responses. The performance after the speed optimization still needs to be tested. Additionally, the Haar-like feature that had been trained in this experiment could not catch 100% of the targets, which suggested more training are necessary to optimize the detector. A better Haar-like detector needed to be trained for boat the datasets in order to outperform the existing OTMACH filter.

The speed optimization could be done with a week or two, and the new ROI detector can be run again with the same dataset to test the speed. Training a better Haar-like kernel may require a larger training set and thus longer machine training time to run. Different setting for the feature selection should be tested to reach an optimized Haar-like kernel. It can be done with two to three weeks. The same datasets can be used to test the detection rate. It will take a few weeks to find a solution that enables the new Haar-like detector to work well with small targets like in the boat datasets.

Personal Comments:

The co-op program provided me an opportunity to continue working on the advanced computer vision project at JPL when I was still in school. The work at JPL was research oriented and my mentor Dr. Lu provided me enough help and advice, while not limiting me to a fixed direction, which gave me a chance to try different techniques and learn at a direction that I was interested the most.

The coordinators Kathy Benjamin and Dr. Francis Flores at the co-op office were very helpful and quick in response.

Acknowledgement:

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, and was sponsored by Research Apprentice program and the National Aeronautics and Space Administration.

BIBLIOGRAPHY

[1] Pisarevsky, Vadim. "OpenCV Object Detection: Theory and Practice," Intel Corporation, Software and Solutions Group. http://fsa.ia.ac.cn/files/OpenCV_FaceDetection_June10.pdf, last visit: April 6, 2010

[2] Johnson, O.C., Edens, W., Lu, T., Chao, T., "Optimization of OT-MACH filter generation for target recognition," Proceedings SPIE Optical Pattern Recognition XX, 7340, (2009).

[3] Viola P., Jones, M., "Rapid Object Detection using a Boosted Cascade of Simple Features," Computer Vision and Pattern Recognition, (2001).

[4] Crow, F., "Summed-area tables for texture mapping", in Processdings of SIGGRAPHY, 18(3):207-212, (1984).

[5] Matas, J. & Sochman, JanZisserman. AdaBoost. The website of Czech Technical University, Prague. Retrieved from http://www.robots.ox.ac.uk/~az/lectures/cv/adaboost_matas.pdf. Last accessed: April 30, 2010.

GLOSSARY

AdaBoost – Adaptive Boosting (of weak classifier). It combines a set of weak classifier (less accurate pattern recognizer) to construct a strong classifier that is more accurate.

ATR – Automated Target Recognition system developed by JPL.

Haar-like feature – A feature extracted from the image in a way similar to Haar filter. It considers the contrast in intensity between two areas in the image.

JPL – Jet Propulsion Laboratory. It is a federally funded research and development center and NASA field center.

NN – Neural network. An artificial neural network composed of artificial neurons (digital simulated). It can be trained by examples to recognize patterns. The output format is usually "true/false", but could also have multiple outputs if needed.

OTMACH – Optimal trade-off Maximum correlation height filter developed by JPL. It uses optical correlation in Fourier domain to find locations in the image that possibly contain target objects. This filter can be trained by examples images.

OpenCV – Open computer vision library developed by Intel. It is a open source project consisting a set of computer vision related library/functions, including Haar-like feature extraction and face detection in images.

ROI – Region of Interest. It is a box marked on the image by the program, which potentially contains an object that needs to be detected.

Sat – Summed area table. It is a technique in summing up the numbers within a rectangle quickly.